Amendments to the Specification:

Please amend the paragraph beginning on page 5, line 18 as follows:

In a preferred implementation, ~~Real Media~~ REALMEDIA is used for the streaming, whereas ~~Real Player~~ RealPlayer® is used as the plug-in player for streaming video and is embedded in a web browser such as Netscape Navigator® or e-Microsoft Internet Explorer®. The program logic is implemented in ~~Javascript~~ JavaScript® code embedded in an HTML page sent from the HTTP server to the client. The browser may advantageously be used as the front end for the IBM CueVideo content-based video and audio retrieval system. This implementation is particularly well suited to on-line browsing of videotaped conferences, allowing users to search, retrieve, and browse particular video segments of interest from a large collection of video clips.

Please amend the section heading on page 6, line 4 as follows:

Brief Description of the ~~Figures~~ Drawings

Please delete the paragraph beginning on page 6, line 13, which is a description of Figure 7.

Please amend the paragraph beginning on page 6, line 15 as follows:

Figure [[8]] 7 is a schematic diagram of an architecture for supporting preferred implementations.

Please amend the paragraph beginning on page 6, line 17 as follows:

Figures [[9]] 8 and [[10]] 9 are look-up tables which facilitate synchronous switching from one media stream to another media stream.

Please add the following paragraph on page 6, beginning on line 19:

Figure 10 shows a computer program product for storing executable program code that may be used in implementing the methods herein.

Please amend the paragraph beginning on page 12, line 7 as follows:

In addition to the video representations discussed above, implementations may include other video representations such as full video at various speeds (such as fast playback or slow motion), a full video at a fixed or variable speed determined by the user, a reverse playback (e.g., fast reverse using the Apple ~~QuickTime Player~~ QUICKTIME PLAYER, as described in U.S. Patent 5,267,334 to Normille et al. entitled "Encoding/decoding moving images with forward and backward keyframes for forward and reverse display" and U.S. Patent 5,739,862 to Cen entitled "Reverse playback of MPEG video"), a low bandwidth (e.g., 50 kbps) representation of the original, a panoramic video (see, for example, "Enhancing distance learning with panoramic video", by J. Foote and D. Kimber, Proceedings of the 34th Hawaii International Conference on System Sciences-2001), a video preview, and a video trailer. "Mosaic" video composed of multiple video tracks can be employed in which either moving storyboards or other kinds of video are employed. In the case of mosaic video, some views may be dedicated to a global view (e.g.,

an entire soccer field), whereas other views may focus in on areas of particular interest (e.g., an individual player on that field). Other media streams may include animation (such as fast moving storyboard without audio) and graphic display (e.g., 3-D graphics). In general, a media stream may be composed of one or more tracks playing in parallel in a synchronous way as a multi-track media stream (as described in, for example, U.S. Patent 5,452,435 to Malouf et al. entitled "Synchronized clocks and media players").

Please amend the paragraph beginning on page 17, line 1 as follows:

Figure 6 is directed to an implementation related to the IBM CueVideo Toolkit, in which an application makes use of SMIL files with ~~Real Player~~ Real Player®. The switching from the first media stream to the second media stream in this case involves additional steps. Because of a limitation in the current ~~Real Player~~ Real Player® Application Program Interface (API), the position in the second media stream, when it is a stream composed using a SMIL file, cannot be set before the second media stream has already started to play. After the first media stream has been stopped (step 98), the ~~Real Player~~ Real Player® is set to the second media stream (step 130). In step 132, the ~~Real Player~~ Real Player® waits for the second media stream to fill the buffer in the ~~Real Player~~ Real Player® and starts to play the second media stream from its beginning. The ~~Real Player~~ Real Player® then pauses the playback (step 134), sets the ~~Real Player~~ Real Player® position in the second media stream to the calculated position (step 136), in which the calculated position has been calculated in step 96. The ~~Real Player~~ Real Player® then begins playing the second media stream at this calculated position (step 138). The logic

inherent in Figure 6 is otherwise similar to that described above in connection with Figures 4 and 5. The user may subsequently select other media streams to view, and the algorithm or a portion of it may be repeated.

Please amend the paragraph beginning on page 17, line 16 as follows:

The Appendix is a section of computer code showing JavaScript® functions for synchronous playback which may be used in a preferred implementation. In particular, the Appendix ~~Figure 7~~ gives JavaScript® code, embedded in a DHTML page, which uses the API of the ~~Real Player~~ Real Player® to implement the functionality described in connection with Figure 6. These JavaScript® functions, which are designed for the client side, provide synchronous playback based on asynchronous streaming video APIs. The different media views (such as those shown in Figure 3) are stored as files on the video server 198, whereas, in this implementation, the browser is implemented as an HTML page that contains ~~Javascript~~ JavaScript® code and an embedded ~~Real Player~~ RealPlayer® object. Although a preferred implementation has been described with respect to ~~Real Player~~ RealPlayer®, other implementations involve the use of players other than ~~Real Player~~ RealPlayer® and may be used with a variety of video, audio and media players.

Please amend the paragraph the paragraph beginning on page 18, line 3 as follows:

A preferred architecture for the implementations disclosed herein is now described, which may be used in either Internet or Intranet applications. Figure [[8]] 7 illustrates architecture for supporting two or three-tier web applications in the context of media streaming. This architecture supports the searching of a media (e.g., video) database by typing in keywords, and retrieving and playing back media clips selected by the user. The user begins with a standard "search" page 170 in Hyper-text markup language (HTML) presented by the client's browser (the Netscape Navigator® or e-Microsoft Internet Explorer® platforms may be used). The page 170 contains a POST form 175 for typing in keywords, which the user then submits by clicking on a "Submit" button. This results in an HTTP request 178 being sent to a web server 182, which in turn calls up an application server 186 . The application server 186 performs the search, and generates the search results as an HTML page 190. The user then selects one or more of the media clips found by the search (each of which preferably has an associated time offset as discussed above in connection with Figure 5), and the media browser allows the user to choose from a number of representations of a given media stream, such as those indicated in Figure 3. The client can play a desired video clip at the appropriate time offset by communicating through a streaming video player plug-in 194 (such as Real Plugin or the IBM Video Charger Plugin, installed at the client side) to a streaming video server 198 (e.g., Real Server or the IBM Video Charger Server). Details regarding the IBM Video Charger Plugin and the IBM Video Charger Server may be found at http://www-4.ibm.com/software/data/videocharger/ .

However, implementations of the invention may be used with a variety of media streaming formats, servers, browsers, and players. The graphical user interface (GUI) functionality used to support playback based on query results is thus extended to support the switching of views from a first media stream (e.g., skim video) to a second media stream (e.g., full length video), starting with the appropriate time offset given by the first media stream at the time of switching.

Please amend the paragraph beginning on page 19, line 4 as follows:

Although the video browsing content can be precomputed and streamed from the video server 198, video control logic (such as calculating time offsets and naming appropriate video source files) may be advantageously generated by a Common Gateway Interface (CGI) program and encapsulated in Dynamic HTML (DHTML) at the client side, as shown in the Appendix ~~Figure 7~~. As is known in the art, CGI is a standard for external gateway programs which permits them to interface with information servers such as HTTP servers. Various hardware and software arrangements for implementing these procedures may be used.

Please amend the paragraph beginning on page 20, line 6 as follows:

One straightforward method of managing time offsets can be appreciated by considering the lookup or time correlation table shown in Figure [[9]] 8. For each video representation (skim video, fast moving storyboard, slow moving storyboard, and full video), each frame in that video representation is listed in the appropriate column using the SMPTE

time format (Society of Motion Picture and Television, see http://www.smpte.org), in which hours, minutes, seconds, and frame number are presented as hh:mm:ss:ff. For each of the views, a frame represents approximately 1/30 second. The rows are arranged so that each entry in a row corresponds to the same point in chronological time, e.g., each of the 4 views within each row in Figure [[9]] 8 would show the same picture. The time correlation table may be computed from analyzing the corresponding media streams, or may be partially or manually entered into the server.

Please amend the paragraph beginning on page 20, line 16 as follows:

The various video representations are tied together, so that 60 seconds of viewing time within the slow moving storyboard view correspond to 15 seconds of viewing time within the fast moving storyboard view and 30 seconds of viewing time within the full video view. The fast moving storyboard view by its very nature is a condensed version of the full video, and consequently, the total number of frames within a fast moving storyboard is less than the total number of frames of the corresponding full video. The opposite holds true for the slow moving storyboard. A skim video representation, on the other hand, reflects only segments of the full video, and hence, there are frames in the full video which do not correspond to frames in the skim video representation, as indicated by the dashes in Figure [[9]] 8. In Figure [[9]] 8, the skim video is seen to begin at a point corresponding to 5 seconds into the full video. The skim video continues for 5 seconds, and then picks up again at a point corresponding to 30 seconds into the full video.

Please amend the paragraph beginning on page 21, line 4 as follows:

If the user decides to move from the slow moving storyboard view to the full video view after viewing the slow moving storyboard for 10 seconds, the lookup table in Figure [[9]] 8 provides the corresponding total viewing time from the start of the full video required to bring the user to this point in the story (5 seconds). A time correlation table may also include information on other kinds of media streams such as the ones disclosed herein, e.g., audio and closed caption text.

Please amend the paragraph beginning on page 21, line 10 as follows:

As an alternative to the lookup table just described, each view (or more generally, each media stream) may be associated with its own table, such as those shown in Figure [[10]] 9, in which separate lookup tables are shown for the fast moving storyboard view and the full video view, respectively. Each table has an index number in the left hand column, which is associated with the viewing time as given by the SMPTE format. The index numbers are chosen such that they allow the different representations to be correlated in a synchronized way, thereby providing a numerical link between different views. The indices thus allow one representation to be mapped into (correlated with) another representation. Within the full video view, n may simply be an integer multiple of the frame number up to that point, as in Figure [[10]] 9. In this example, 30 seconds of viewing time within the full video view (00:00:30:00), is given by the index 1800, which is seen to correspond to 15 seconds of viewing time within the fast moving storyboard view (00:00:15:00).

Please amend the paragraph beginning on page 23, line 5 as follows:

Further, synchronization data like that shown in Figure [[10]] 9 may be streamed with the media stream itself to facilitate the necessary calculations. In this case, it may be advantageous for the video server 198 to continually update the lookup table data as the video corresponding to the currently played media stream is viewed. This can be accomplished through the use of an additional data track dedicated to streaming lookup table data: such data may then be used in calculations related to one or more media streams, e.g., moving storyboard, audio, closed caption text, etc.

Please add the following several pages of code (which formerly appeared as Figure 7) as an Appendix to appear at the end of the Specification on page 24:

APPENDIX

```
<HTML>
<HEAD>
<TITLE>IBM CueVideo Synchronized MultiMedia Browser</TITLE>
<meta name="SOURCE" value="IBM CueVideo">
<meta name="TITLE" value="IBM CueVideo Synchronized Media Browser">
<meta name="COPYRIGHT" value="Copyright (c) 2000 by IBM Corporation. All Rights
Reserved.">
</HEAD>
<SCRIPT LANGUAGE="JavaScript">
PLAYER_SYNCHTIME = 10;
NumOfHits= 2;
CurntHit= 1;
FVStartTimes = new Array(2);
BaseNames = new Array(2);
TitleNames = new Array(2);
MatchsNames = new Array(2);
ViewSuffix = new Array(5);
myTimer = null;
CurntPos=0;
CurntRate=1.0;
CurntSource=null;
CurntOrigLength=0;
CurntMode=1;
can_play = false;
RealRoot='http://cvideo.almaden.ibm.com:8080/ramgen/cuelearn/';
HttpRoot='http://cvideo.almaden.ibm.com/cuelearn/';
bn = '                    ';

function InitVars() {
ViewSuffix[0]="";
ViewSuffix[1]="";
ViewSuffix[2]="-msb";
ViewSuffix[3]="-msb_0.666";
ViewSuffix[4]="-msb_1.25";
BaseNames[0]='coffee';
TitleNames[0]='<U>coffee</U> [00:07:37] ';
MatchsNames[0]='Matched words:
<U><B>warm</B></U>    Score:<U>100.00</U> ';
```

```
FVStartTimes[0] = 457000.0;
BaseNames[1]='coffee';
TitleNames[1]='<U>coffee</U> [00:05:43] ';
MatchsNames[1]='Matched words:
<U>warm</U>    Score:<U>69.44</U> ';
FVStartTimes[1] = 343000.0;
}

function CheckPlugin() {
var mimetype = 'audio/x-pn-realaudio-plugin';
if (navigator.appName == 'Netscape') {
if (navigator.mimeTypes[mimetype] != null) {
if (navigator.mimeTypes[mimetype].enabledPlugin != null) {
can_play = true;
}
}
}
else
{
if (document.embeds) can_play=true;
}
}

function PositionControls() {
CheckPlugin();
if (!can_play)
{
alert('To use this web site please install the free RealPlayer 7 Basic plug-in.');
window.location = '/the_plugin_version_update_page.html';
}
else
{
InitVars();
PositionTitle();
PositionViews();
var foo=document.Video.GetVersionInfo();
var foobar=parseFloat(foo);
if (foobar<6){
alert('To use this web site please upgrade your plug-in to the free RealPlayer 7.');
window.location = '/the_plugin_version_update_page.html';
}
```

```
VideoName =
RealRoot+BaseNames[CurntHit-1]+'_course/video/'+BaseNames[CurntHit-1]+ViewSuffix[Curn
tMode]+'.smil';
PlayFullVideo(VideoName, FVStartTimes[CurntHit-1]*CurntRate);
}
}
function RepositionControls() {
PositionTitle();
PositionViews();
VideoName =
RealRoot+BaseNames[CurntHit-1]+'_course/video/'+BaseNames[CurntHit-1]+ViewSuffix[Curn
tMode]+'.smil';
PlayFullVideo(VideoName, FVStartTimes[CurntHit-1]*CurntRate);
}

function PositionTitle() {
Title = '';
Title += 'Query: <U>warm</U>    ';
 Title += ' Result '+CurntHit.toString()+'/'+NumOfHits.toString()+': ';
Title += TitleNames[CurntHit-1]+'<BR>';
Title += MatchsNames[CurntHit-1];
var output=
'<table width="640" border="0" cellspacing="5" cellpadding="2">'+
'<tr> '+
'<td><img src="/images/cvideo_banner.gif"  alt="IBM CueVideo" border="0"></td>'+
'</tr>'+
'<tr bgcolor="#21249C"> '+
'<td> <font color="#FFFFFF" face="Helvetica, HELV,
sans-serif">'+Title+'</font></td>'+
'</tr>'+
'</table>';
if (document.all)
 document.all('TitleArea').innerHTML = output;
else if (document.layers) {
 document.layers['TitleArea'].top=5;
 document.layers['TitleArea'].left=5;
 document.layers['TitleArea'].width=800;
 document.layers['TitleArea'].height=200;
 document.layers['TitleArea'].document.open();
 document.layers['TitleArea'].document.writeln(output);
 document.layers['TitleArea'].document.close();
}
}
```

```
function PositionViews() {
bn=BaseNames[CurntHit-1];
MSBName  = RealRoot+bn+'_course/video/'+bn+'-msb.smil';
FMSBName = RealRoot+bn+'_course/video/'+bn+'-msb_0.666.smil';
SMSBName = RealRoot+bn+'_course/video/'+bn+'-msb_1.25.smil';
ANMName  = RealRoot+bn+'_course/video/'+bn+'fixed.smil';
SB2DName = HttpRoot+bn+'_course/images/'+bn+'-sb0001.html';
VideoName= RealRoot+bn+'_course/video/'+bn+'.smil';

 var output= '<BR><FORM METHOD=POST>'+
'<TABLE NOBORDER CELLSPACING=5 CELLPADDING=5 WIDTH="640" >'+
'<TR><TD ALIGN=center >'+
'<INPUT TYPE="button" Value="Storyboard (SB)"
onClick="location.href=\"+SB2DName+'\"">'+
'<INPUT TYPE="button" Value=" Moving SB "
onClick="JumpToVideo(\"+MSBName+'\',2,1.0)">'+
'<INPUT TYPE="button" Value="  Fast SB  "
onClick="JumpToVideo(\"+FMSBName+'\',3,0.666)">'+
'<INPUT TYPE="button" Value="  Slow SB  "
onClick="JumpToVideo(\"+SMSBName+'\',4,1.25)">'+
'<INPUT TYPE="button" Value="  FullVideo  "
onClick="JumpToVideo(\"+VideoName+'\',1,1.0)">';

output=output+'</TD></TR>';

if (NumOfHits>1)
{
output = output +
'    <TR><TD ALIGN=center>'+
'    <FONT COLOR=WHITE> '+
CurntHit.toString()+' of '+NumOfHits.toString()+ '  '+
'<INPUT TYPE="BUTTON" VALUE="Prev Result" onClick="ViewPrev()";>'+
'<INPUT TYPE="BUTTON" VALUE="Next Result" onClick="ViewNext()";></TD></TR>';
}
else
{
output = output +
'    <TR><TD ALIGN=right>'+
'    <FONT COLOR=BLACK>  '+'A '+' </font>'+
'</TD></TR>';
}
```

```
output=output+'</table>';
if (document.all)
document.all('ViewsTable').innerHTML = output;
else if (document.layers) {
document.layers['ViewsTable'].top=350;
document.layers['ViewsTable'].left=10;
document.layers['ViewsTable'].width=600;
document.layers['ViewsTable'].height=200;
document.layers['ViewsTable'].zIndex=40;
document.layers['ViewsTable'].document.open();
document.layers['ViewsTable'].document.writeln(output);
document.layers['ViewsTable'].document.close();
}
}

function PlayFullVideo(s,i) {
if (s!=CurntSource) {
  if (CurntSource!=null)
    document.Video.DoStop();
document.Video.SetSource(s);
CurntSource=s;
}
else if (CurntOrigLength>0) {
  ThisLength=document.Video.GetLength();
  i=i*ThisLength/CurntOrigLength;
}
CurntPos = i;
if (document.Video.GetPlayState() != 3) {
document.Video.SetVolume(100);
document.Video.DoPlay();
 if (CurntPos > 1000)
myTimer=setTimeout('WaitForFirstPlay()',PLAYER_SYNCHTIME);}
else {
if (CurntPos>0) {
document.Video.DoPause();
document.Video.SetPosition(CurntPos);
document.Video.SetVolume(100);
document.Video.DoPlay();
}
}
}
```

```
function WaitForFirstPlay () {
clearTimeout(myTimer);
if (document.Video.GetPlayState() != 3)
myTimer=setTimeout('WaitForFirstPlay()',PLAYER_SYNCHTIME);
else
{
document.Video.DoPause();
if (CurntMode==1)
  CurntOrigLength=document.Video.GetLength();
document.Video.SetPosition(CurntPos);
document.Video.DoPlay();
}
}

function WaitForPlay () {
clearTimeout(myTimer);
if (document.Video.GetPlayState() != 3)
myTimer=setTimeout('WaitForPlay()',PLAYER_SYNCHTIME);
else
{
document.Video.DoPause();
theoffset = CurntPos*CurntRate;
document.Video.SetPosition(theoffset);
document.Video.DoPlay();
}
}

function JumpToVideo(newvideo,newMode,newRate) {
CurntPos = document.Video.GetPosition()/CurntRate;
document.Video.DoStop();
document.Video.SetSource(newvideo);
CurntMode=newMode;
CurntRate=newRate;
document.Video.DoPlay();
if (CurntPos > 1000)
myTimer=setTimeout('WaitForPlay()',PLAYER_SYNCHTIME);
}
```

```
function ViewNext() {
if (CurntHit<NumOfHits)
CurntHit=CurntHit+1;
else
CurntHit=1;
//update StartSB time
RepositionControls();
}

function ViewPrev() {
if (CurntHit>1)
CurntHit=CurntHit-1;
else
CurntHit=NumOfHits;
//update StartSB time
RepositionControls();
}


</SCRIPT>

<BODY TEXT="#000000" LINK="#3333FF" VLINK="#3366FF" ALINK="#FFFF00">
<BODY bgcolor="#000000" >
<BODY onLoad=PositionControls() >
<BR>
<SPAN ID="TitleArea"
STYLE="top:5;left:5;height:200;width:800;position:absolute"></SPAN>
<SPAN ID="ViewsTable"
STYLE="top:370;left:5;width:600;height:200;position:absolute"></SPAN>
<BR><BR><BR><BR><BR><BR>
<table border=0 cellpadding="0" cellspacing="0" bgcolor="#000000" width="640" >
<tr>
<td align=center>
<EMBED NAME=Video SRC="http://cvideo.almaden.ibm.com:8080/ramgen/cuelearn/?embed"
WIDTH=600 HEIGHT=200

CONTROLS=ImageWindow CENTER=true CONSOLE=_master><BR>
</td>
</tr>
<tr>
<td align=center>
<EMBED SRC="http://cvideo.almaden.ibm.com:8080/ramgen/cuelearn/?embed" WIDTH=40
```

HEIGHT=20 CONTROLS=PlayButton CONSOLE=_master>
<EMBED SRC="http://cvideo.almaden.ibm.com:8080/ramgen/cuelearn/?embed" WIDTH=300

HEIGHT=20 CONTROLS=PositionSlider CONSOLE=_master>
<EMBED SRC="http://cvideo.almaden.ibm.com:8080/ramgen/cuelearn/?embed" WIDTH=100

HEIGHT=20 CONTROLS=PositionField CONSOLE=_master>

</td>
</tr>
</table>
</BODY>
</HTML>